Kennesaw State University
Analytics and Data Science Institute

Improved Regional Sports Scheduling using SAS PROC OPTGRAPH
Andrew Henshaw

Faculty Advisor: Dr. Joe DeMaio

INTRODUCTION

This project improves upon the manual clustering method of regional sports scheduling used by Georgia Soccer and most other state organizations. Commonly, schedules are formed by manually grouping teams by visual inspection of geographic location. With this approach, *n* teams are assigned to a cluster and each team plays all of the others once per season (*n*-1 games). This scheme is simple to implement, but, unfortunately, each group is treated as being completely isolated. Therefore teams at the boundaries of neighboring cliques may be natural opponents, but would not be scheduled against each other.

The proposed system uses travel time between venues as the edge weight and multiple rounds of linear-assignment mapping to generate a full schedule. Two alternatives of this approach are considered: a multi-round, bipartite, minimum-weight matching algorithm implemented with SAS PROC OPTGRAPH and a non-bipartite version implemented in Python.

METHODS

To compare the three alternatives, the B16 division from the Fall 2017 season was considered. With 78 teams, this was the largest division for that season. Travel times between each team's home venue were obtained using the Google Maps API. Figure 1 shows the pairings for teams generated using the manual clustering and round-robin assignment approach. Figures 2 and 3, respectively, illustrate the pairings generated by the bipartite and the non-bipartite algorithms.

A formally-optimal solution to this problem is not possible [Ribeiro], however a round-by-round, minimum-weight matching is a very good heuristic approach. As PROC OPTGRAPH only supports bipartite matching (using the Munkres algorithm [Galati]), a work-around was developed. Each round, the teams are evenly and randomly split into two sets. The matching is done and the weights for the paired teams are changed to infinity. This process is repeated until the desired number of rounds have been scheduled (Listing 1).

The Python implementation is able to use a less well-known algorithm [Galil] that supports non-bipartite matching; therefore, no splitting of teams is required. Both implementations optimize for the total drive-time for all games of all teams in a division.

```
/* choose best matching for this round */
proc optgraph
    graph_direction = directed
    data_links      = bipartite;
    data_links_var
              from   = home
              to     = away
              weight = weight ;
    linear_assignment out = pairings;
run;

/* for all pairs in this round, increase weight so that
   the pair is not re-selected */
proc sql;
    update weights
    set weight=1000000
    where exists (
       select * from pairings
       where (pairings.home = weights.home and pairings.away = weights.away)
       or    (pairings.home = weights.away and pairings.away = weights.home)
    );
quit;
```

Listing 1: SAS Macro Fragment

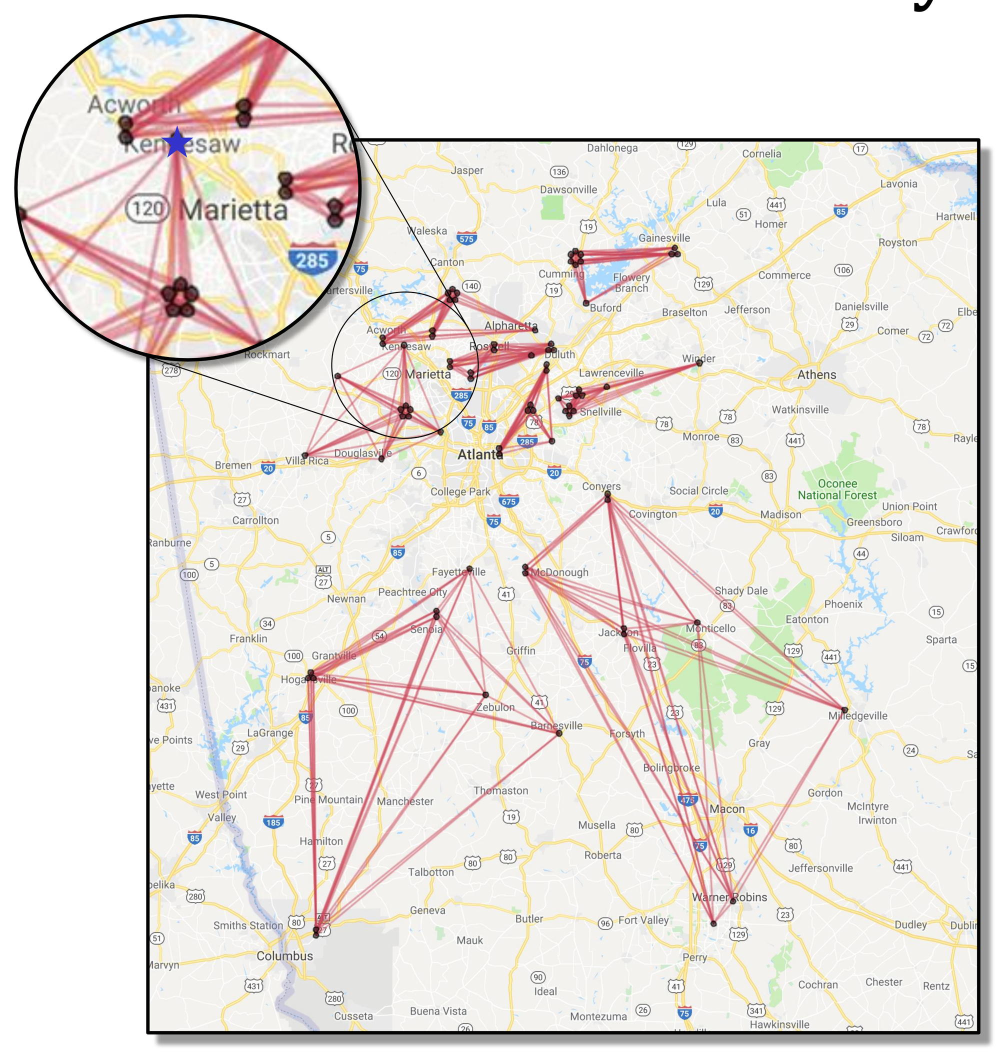


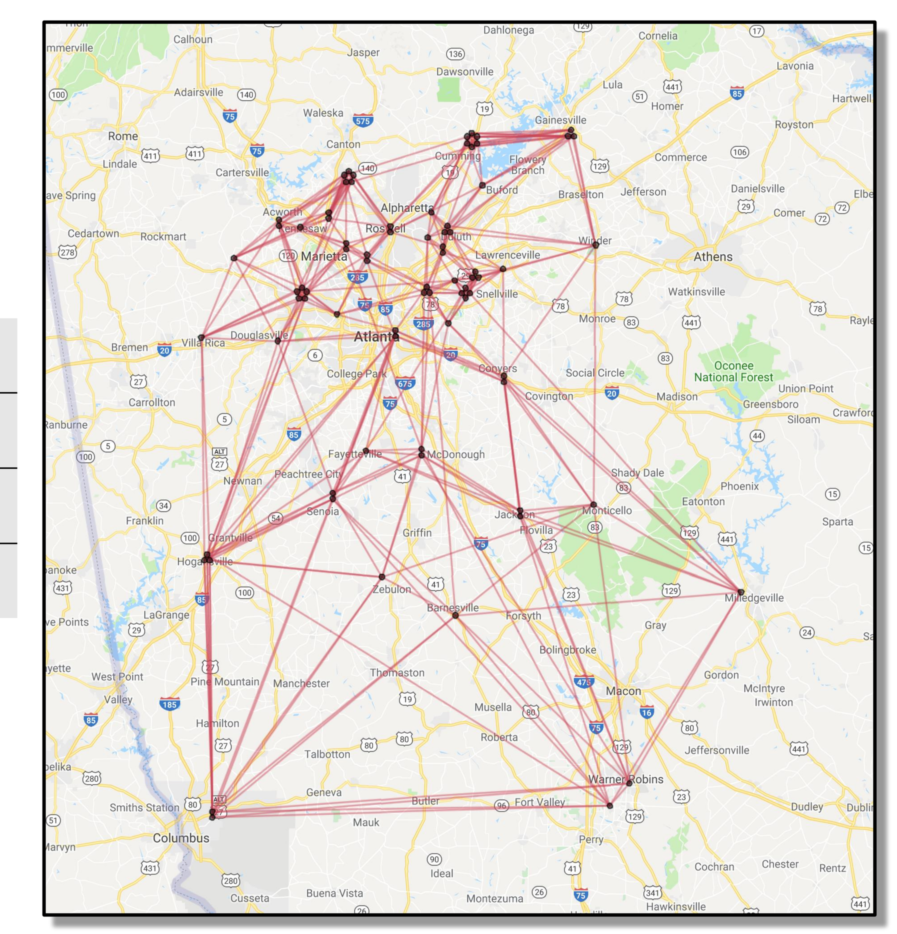
Figure 1
Conventional Scheduling with Manual Clustering, Cliques, and Round-Robin Pairing

78	Number of Teams
351	Number of Games
14,840 mi.	Aggregate Round-Trip Distance
340 hrs.	Aggregate Round-Trip Drive Time

Figure 2
Multi-Round, Bipartite, Minimum-Weight Matching (SAS PROC Optgraph)

Number of Teams	78
Number of Games	351
Aggregate Round-Trip Distance	14,710 mi.
Aggregate Round-Trip Drive Time	330 hrs.

–2.9%



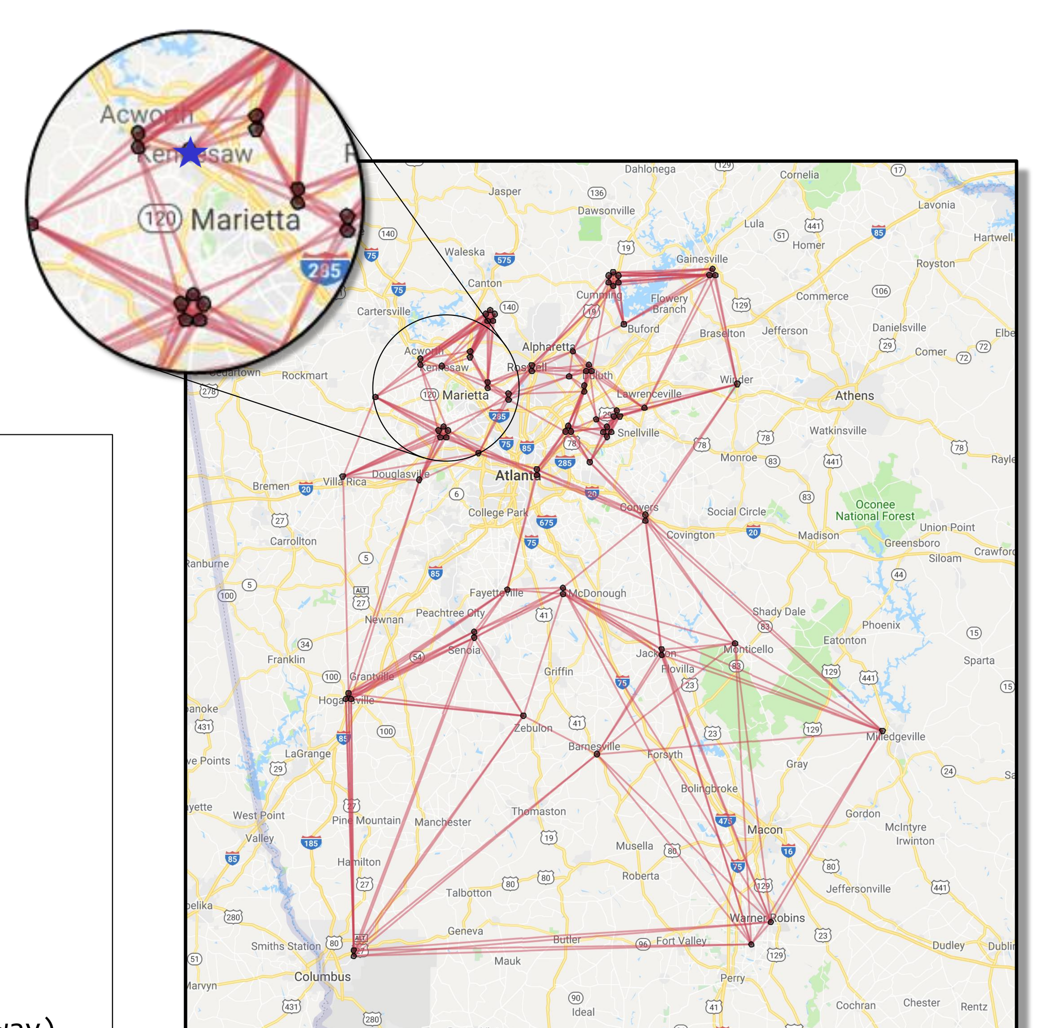


Figure 3
Multi-Round, Non-Bipartite, Minimum-Weight Matching

78	Number of Teams
351	Number of Games
13,122 mi.	Aggregate Round-Trip Distance
303 hrs.	Aggregate Round-Trip Drive Time

–12.3%

FINDINGS

As shown in Figure 3, the use of a multi-round, non-bipartite minimum weight matching algorithm provides a substantial improvement in performance (**12.3% reduction in drive time**). In addition, this approach eliminates the issue of the non-pairing of natural neighbors caused by the use of isolated clusters.

Although better than the manual method, the bipartite solution, as expected, suffered from the incomplete options that were forced by having to artificially isolate the teams into disjoint sets. Randomly splitting the teams between groups after each round ameliorates the issue, but the results are still inferior to the non-bipartite algorithm.

CONCLUSIONS

The Multi-Round, Non-Bipartite, Minimum-Weight Matching algorithm has been successfully incorporated into a complete scheduling solution for Georgia Soccer for **two seasons** (one year). Based upon this analysis, we can see that the use of this new system has already provided the following benefits:

- reduced round-trip drive distance by **4.9 miles per game**
- reduced round-trip drive time by over **6 minutes per game.**

Each year, the new system can be expected to:

- reduce aggregate trip distance by **12,600 miles**
- save each participant nearly **1 hour** of drive time (on average).

Assuming new car fuel efficiency and 8 cars per team per trip, the program will:

- reduce total drive distance by **101,000 miles per year**
- reduce fuel consumption by **4300 gallons per year**
- save nearly **$10,000 per year** in fuel costs.

REFERENCES

Galati, M., & Liao, Y. (2013). Linear Assignment Statement. *SAS OPTGRAPH Procedure 12.3: Graph Algorithms and Network Analysis*, 34-35.

Galil, Z. (1986). Efficient algorithms for finding maximum matching in graphs. *ACM Computing Surveys*, 18, 23-38.

Goossens, D. R., & Spieksma, F. C. R. (2012). Soccer schedules in Europe: An overview. *Journal of Scheduling*, 15(5), 641–651.

Ribeiro, C. C. (2012) Sports scheduling: Problems and applications. *International Transactions in Operational Research,* 19, 201-226.

Toffolo, T.A.M., Christiaens, J., Spieksma, F.C.R. et al. (2017). The sports team grouping problem. *Annals of Operations Research,* 1-21.

ACKNOWLEDGEMENTS

Grateful acknowledgements to all of the staff at Georgia Soccer for their dedication to the sport of soccer, to Dr. Jennifer Priestley and Dr. Sherrill Hayes for their academic support, and to Dr. Joe DeMaio for his guidance in tackling this problem.